[19] 中华人民共和国国家知识产权局

[51] Int. Cl.
*G06F 9/30 (2006.01)*
*G06F 9/32 (2006.01)*

[12] 发明专利申请公布说明书

[21] 申请号 200810173125.9

[43] 公开日 2009 年 3 月 18 日

[11] 公开号 CN 101387951 A

[22] 申请日 2005.3.30
[21] 申请号 200810173125.9
分案原申请号 200510076224.1
[30] 优先权
[32] 2004.3.31 [33] US [31] 10/815478
[71] 申请人 阿尔特拉公司
地址 美国加利福尼亚州
[72] 发明人 J·L·巴尔

[74] 专利代理机构 中国专利代理(香港)有限公司
代理人 王洪斌 魏 军

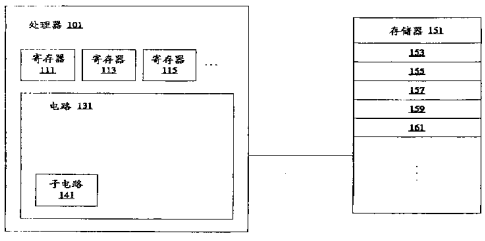
权利要求书 2 页 说明书 13 页 附图 10 页

[54] 发明名称

优化的处理器和指令对准

[57] 摘要

本发明涉及优化的处理器和指令对准。一种方法和设备被提供用来优化处理器核心。公共的处理器子电路被用来执行对各种不同类型指令的计算，包括转移和非转移指令。即使被支持指令是多字节或字对准的，增加跨越不同指令类型的计算的通用性也允许转移指令跳转到字节对准的存储器地址。



1. 一种现场可编程门阵列，包括：

多个寄存器；

被配置成用以处理与包括多个转移指令和非转移指令的指令集相关联的多个指令的电路，所述多个指令均具有多字节的长度，所述多个指令能够被在多字节对准的地址访问；

公共子电路，可操作来执行非转移指令中的立即字段的符号扩展以及执行转移指令中的所述立即字段的符号扩展以计算转移指令的目标地址，其中，对所述非转移指令进行操作的所述公共子电路是对所述转移指令进行操作的相同的子电路；

其中，所述现场可编程门阵列的主元件或者从元件中的一个直接通过端口来访问所述阵列的存储器而不是通过系统总线来访问。

2. 如权利要求 1 所述的现场可编程门阵列，其中，所述阵列不包括系统总线。

3. 如权利要求 1 所述的处理器，其中，所述多个指令被在字对准的地址访问。

4. 如权利要求 1 所述的处理器，其中，所述多个指令被在半字对准的地址访问。

5. 如权利要求 1 所述的处理器，其中，转移指令包括转移偏移量和当前程序计数器值。

6. 如权利要求 1 所述的处理器，其中，转移偏移量和当前程序计数器的单位是字节。

7. 如权利要求 1 所述的处理器，其中，所述多个指令的长度是一个字。

8. 一种处理器，包括：

多个寄存器；

被配置成用以处理与指令集相关联的多个转移指令和非转移指令的电路，所述多个转移指令和非转移指令包括立即字段；和

公共子电路，执行与一个或者多个转移指令相关联的立即字段的符号扩展以及执行与一个或者多个非转移指令相关联的所述立即字段的符号扩展，其中，

对所述非转移指令进行操作的所述公共子电路是对所述转移指令进行操作的相同的子电路，其中，与一个或者多个转移指令相关联的所述立即字段的所述符号扩展被执行以确定转移目标地址；和

其中，所述现场可编程门阵列的主元件或者从元件中的一个直接通过端口来访问所述阵列的存储器而不是通过系统总线来访问。

9.如权利要求 8 所述的处理器，其中，所述指令集包括多个指令。

10.如权利要求 9 所述的处理器，其中，所述多个指令被在半字对准的地址访问。

11.如权利要求 8 所述的处理器，其中，转移指令包括转移和条件转移指令。

12.如权利要求 8 所述的处理器，其中，公共子电路被用于处理与所述转移和非转移指令相关联的所述立即字段，其中，立即字段值以字节为单位而被保持。

13.如权利要求 12 所述的处理器，其中，公共子电路被用于执行与所述转移和非转移指令相关联的所述立即字段的符号扩展。

# 优化的处理器和指令对准

本申请是分案申请，其母案申请日是2005年3月30日，其母案申请号是200510076224.1，其母案发明名称：优化的处理器和指令对准。

## 技术领域

本发明涉及处理器。在一个例子中，本发明涉及用于优化处理器和提供灵活的指令对准的方法和设备。

## 背景技术

常规处理器被配置来支持多种指令。在许多情况中，处理器电路被配置来执行特定的指令。指令往往用特别配置的子电路来依次处理。在一种情况中，一种特定类型的指令，转移指令，允许通过在转移目标地址跳转到非顺序指令来改变程序执行流。专门的电路往往被提供来计算转移指令跳转操作的目标地址。

然而，有效地执行用于不同类型指令的计算机理是有限的。在一个例子中，专门的子电路被配置来特别地计算多字节对准的转移目标地址。然而，专门的子电路往往不被再用于其它类型的指令。

用于有效优化处理器核心的机理是有限的。因此，希望提供改进的方法和设备以用于优化处理器和处理器电路的实现。在某些情况下，对准的限制可以被放宽来进一步优化处理器。

## 发明内容

用于优化处理器核心的方法和设备被提供。公共的处理器子电路被用来执行对各种不同类型指令的计算，包括转移和非转移指令。即使被支持指令是多字节或字对准的，增加跨越不同指令类型的计算的通用性也允许转移指令跳转到字节对准的存储器地址。

在一个实施例中提供了一个处理器。该处理器包括多个寄存器和电路。电路被配置来处理与包括转移和非转移指令的指令集相关的指令。每个指令都具

有多字节的长度。指令在多字节对准的地址是可访问的。多字节对准的转移指令可操作来访问在字节对准的地址中的指令。

在另一个实施例中提供了一个处理器。该处理器包括多个寄存器和电路。电路被配置来处理与指令集相关的转移和非转移指令。转移指令和非转移指令包括一个立即字段(immediate field)。公共的子电路被用来处理与一个或多个转移指令和一个或多个非转移指令相关的立即字段。

在另一个实施例中提供了一个用于执行指令的方法。与地址相关的转移指令被解码。转移指令具有一个相关的操作码和一个立即值(immediate value)。转移目标地址用立即值来计算。转移目标地址通过使用公共子电路来确定。公共子电路可操作来计算字节对准的地址。公共子电路还被配置来执行非转移操作。跳转被执行到转移目标地址。转移目标地址是多字节对准的。

本发明的这些及其它特征和优点将在以下本发明的说明和附图中更详细地给出，其通过举例的方式说明了本发明的原理。

附图说明

本发明可以通过参考结合附图的下列说明来最好地理解，其说明本发明的特定实施例。

图 1 是一个处理器的示意图。

图 2 是非转移指令的一个示意图。

图 3 是转移指令的一个示意图。

图 4 是使用公共子电路来执行的非转移指令的一个示意图。

图 5 是使用公共子电路来执行的转移指令的一个示意图。

图 6 是指令处理的一个流程图。

图 7 是可编程芯片的一个示意图。

图 8 是互连架构(interconnection fabric)的一个示意图。

图 9 是用于实现可编程芯片的技术的一个示意图。

图 10 是描述计算机系统的一个示意图。

具体实施方式

现在将详细说明本发明的一些特定实施例，其中包括发明人构思用于执行

本发明的最佳模式。这些特定实施例的例子在附图中被说明。虽然本发明结合这些特定实施例被描述，然而应当理解这不意味着将本发明限制于这些所述的实施例。相反，其意图覆盖包括在附加权利要求所定义的本发明的精神和范围内的替换、修改和等效物。例如，本发明的技术将在特定的处理器和存储器环境中来描述。

然而应当指出，本发明的技术可以被应用于各种类型的设备。在下列说明中，为了彻底理解本发明而阐明了大量的细节。本发明可以脱离所有的这些细节或其一部分而被实践。在其它情况中，为了避免不必要地模糊本发明而对熟知的处理操作不作详细描述。此外为了清楚起见，本发明的技术和机理有时将用单一的形式来描述。然而应当指出，除非另外提示，一部分实施例可以包括技术的多个反复操作或机理的多个实例化。例如，处理器被用于各种环境。然而应当理解，在本发明的范围内也可以使用多个处理器。

常规处理器被配置来支持多种指令。一些处理器支持通用指令集，比如精简指令集计算(RISC)指令集、复杂指令集计算(CISC)、或超长指令字(VLIW)指令集。其它诸如数字信号处理(DSP)处理器或视频加速器之类的更专门的处理器可以支持更专门的指令集。典型的处理器将基于指令类型和接收的相关联的参数来执行操作。例如，处理器可以执行随后是诸如载入和存储之类的存储器存取指令的加法指令序列。

指令集可以包括各种各样的指令，其中包括诸如无符号加法或无符号减法之类的算术指令，诸如逻辑与和逻辑或之类的逻辑指令，诸如载入字或存储字之类的数据传送指令，诸如等于则转移或不等于则转移之类的转移指令以及跳转指令。一些处理器体系结构允许指令集中的指令改变长度。例如，一些指令可能是 16 位长度的指令，而其它的指令是 32 位长度。然而，其它体系结构使用具有固定长度的指令。使用只包括固定长度指令的指令集极大地简化了指令集的实现。

简单性在许多情况中产生效率。因为指令是固定长度的，所以指令可以用特定的存储器地址来对准。例如，指令存储器以字节单元来编址。在任何字节地址都具有起始地址的任何指令在此称之为字节对准的指令。具有起始地址是字节地址大于 1 的整数倍数的任何指令在此称之为多字节对准的指令。例如，多字节对准的指令的特定情况是字对准的指令。字对准的指令具有每隔四个字

节对准的起始地址。

从而，指令存储器中的任何字对准的指令都将起始于 4 字节、8 字节、12 字节、16 字节等等的地址。因为指令是字对准的，所以诸如转移指令之类的一些指令只能跳转到特定的字对准的地址。诸如等于则转移或不等于则转移指令之类的转移指令允许通过跳转到在转移目标地址处的非顺序指令来改变程序的执行流。在某些例子中，转移目标地址作为嵌入在转移指令中的偏移和转移指令地址的函数来计算。在其它例子中，转移指令可能是一个给定的绝对地址。常规处理器使用专门的子电路来计算转移目标地址。专门的子电路通过假定转移目标地址是多字节对准的或字对准的来确定转移目标地址。

本发明的技术和机理认识到，使用专门的子电路来确定多字节对准的或字对准的转移目标地址是低效率的。通过增加跨越不同指令类型的计算通用性，本发明的技术和机理最小化硬件成本并最大化性能。例如，用于执行载入或存储指令的子电路也被用于执行转移指令。通过使用公共的子电路，硬件成本由于相同的子电路可以跨越不同的指令类型被再用而被减少。对于非公共的计算不需要使用专门的子电路。减少处理器中的电路数量通常允许增加处理器频率和性能。

在一个例子中，本发明的技术允许转移目标的有效计算。在典型的情况中，计算转移目标涉及确定与转移指令相关的立即字段的符号扩充(sign extended)值。处理器已经具有其它的使用相同立即字段的符号扩充值来执行计算的非转移指令。然而，多字节对准的问题通常阻止使用相同的子电路来执行两种类型的指令。通过选择将以字节出现的转移偏移，用于对非转移指令执行立即字段的符号扩充的相同子电路可以被再用于转移指令。

允许子电路再用于不同的指令类型同时也有其它的好处。通过再用子电路，提供了指令集的变化支持。使用本发明的技术和机理，新的指令可以被增加并得到灵活的处理。例如，当诸如 16 位指令之类的较小指令被引入 32 位指令集时，对准的限制一般被简化来匹配最小指令的大小。使用本发明的技术和机理，即使新的指令不是多字节对准的，转移指令也能够跳转到这些指令。例如，如果指令集中的指令先前被字对准，新指令可以是字节对准或者半字对准，并且转移指令将仍然能够跳转到适当的存储器单元。

如果不使用本发明的技术，则现存的转移指令将不得不被改变从而破坏兼

容性，或者新的转移指令将不得不被增加以支持较小指令的对准的限制。通过增加一整套新的转移指令，不得不引入的附加的子电路，潜在地增加了硬件成本。一些常规的处理器使用多字节对准的指令。许多常规处理器都被配置来支持较小的指令。在许多情况下，新的转移指令被简单写入来处理这些特定的较小指令。常规的处理器从而与不同的转移指令群放在一起。一个转移指令群处理 16 位对准的指令。另一个类似的转移指令群处理 8 位对准的指令。常规处理器中的这些多余的转移指令群效率非常低。

图 1 是被配置来处理指令集的一个处理器示意图。处理器 101 包括有寄存器 111、寄存器 113 和寄存器 115 的寄存器组。处理器 101 还包括用于处理各种不同指令的电路 131。电路 131 包括子电路 141。用于执行指令集中指令的处理器电路的任何部分在此称之为子电路。处理器 101 被连接到具有存储器行(memory line)153-161 的存储器 151。应当指出，处理器 101 还可能具有其它配置并包括其它未示出的元件，比如与指令高速缓存、数据高速缓存、以及存储器控制器。在典型的例子中，处理器依次从存储器读取指令并执行这些指令。指令可能涉及对数据进行操作，该数据也是从存储器载入并被放置于寄存器中以用于更加有效的访问。执行诸如算术操作之类的操作涉及到使用处理器 101 内特定的硬件。转移指令可能涉及使用处理器 101 内的另一个硬件块。

在一个极度简化的例子中，每个指令用处理器 101 内的不同子电路来处理。例如，逻辑“与”操作可以使用子电路 141、寄存器 111 以及指令中提供的特定值而被执行。指令中用于提供特定操作的执行信息的参数在此称之为立即值。在某些情况下，立即值在先前的指令中从存储器中获得。

根据本发明的各种不同实施例，转移指令允许处理操作序列来移动到不同的多字节对准的存储单元。在一个例子中，指令是字对准的。转移指令允许转移到特定的字对准的地址。然而不允许转移到其它的地址。应当理解，通常对多字节或字对准的存储单元的限制对准允许访问范围更广的存储器地址。更特别地，有限位数可以用来存储与转移相关的偏移。在一个简化例子中，偏移的 3 位或八个不同值被提供。如果没有使用多字节对准，则转移目标地址在存储空间中被限制在三比特的八个字节的范围中。然而，如果使用了字对准，则偏移被乘以 4。转移目标地址的范围现在被扩大为 5 位的范围或存储空间中的 32 个字节。在这个例子中，目标地址仍然只能在 18 个存储单元中，然而代替被限于

8 字节的范围，8 个特定的存储单元被遍布 32 个字节的范围。

从而，许多处理器结构使用多字节对准来扩充转移目标地址的范围。包括转移指令的潜在目标地址的任何地址范围被称为转移目标地址范围。在许多处理器和专用集成电路中，处理唯一指令的加法逻辑和子电路比在可编程芯片中处理唯一指令的加法逻辑更廉价。许多可编程器件中的逻辑很受重视，因此本发明的技术认识到即使转移目标地址的范围被减小，再用子电路以用于处理不同类型的指令也是特别有益的。

图 2 是常规指令的一个示意图。根据各种不同的实施例，指令指定转移指令 201 的操作码、空字段 203 和 205、以及立即值 207。根据各种不同的实施例，对于指令中 32 比特的总数，用于转移指令 201 的操作码是 6 个比特，空字段 203 和 205 每个都是 5 个比特，而立即字段 207 是 16 个比特。根据各种不同的实施例，指令是字对准的。当处理器到达特定的转移指令时，它确定使用转移目标存储器地址应该被找到的偏移。在一个例子中，通过将转移偏移乘以 4 来计算转移目标地址，从而确保字对准和增加转移目标地址范围。计算结果然后被符号扩充到指令地址中的比特数。符号扩充值然后被添加到转移指令地址或程序计数器。在一个例子中，计算如下所示：

转移目标=程序计数器+符号扩充(立即字段*4)

因为转移偏移的单位是字，而且程序计数器和转移目标的单位是字节，所以立即值被乘以 4。

图 3 是常规算术指令的一个示意图。根据各种不同的实施例，指令包括整数加操作码(opcode)301、源寄存器 303、目的寄存器 305、和立即字段值 307。根据各种不同的实施例，对于指令中 32 比特的总数，用于整数加指令 301 的操作码是 6 比特，寄存器字段 303 和 305 每个是 5 比特，而立即字段 307 是 16 比特。在一个例子中，整数加指令取在源寄存器字段 303 中识别的寄存器值，将其添加到立即字段 307 的符号扩充值，并将计算结果写入目的寄存器字段 305 所指定的寄存器。在一个例子中，立即字段从 16 比特被符号扩充到 32 比特，或者说是一个指令长度。根据各种不同的实施例，整数加指令被计算如下：

目的寄存器值=源寄存器值+符号扩充(立即字段)

尽管转移指令和整数加指令对立即字段值执行符号扩充操作，然而转移指令在执行符号扩充之前还是将立即字段值乘以 4。乘法运算被执行以便允许字对

准和扩充的目标转移地址范围。然而，通过要求在符号扩充之前执行乘 4 的乘法运算，需要不同的子电路布线(route)来执行该操作。在许多情况中，一个子电路被特别配置用于符号扩充立即字段，另一个子电路被特别被配置用于符号扩充乘以 4 的立即字段。在某些例子中，乘以 4 的乘法运算通过在基础的符号扩充子电路上使用附加的多路复用器和移位器来执行。然而，附加硬件或附加子电路的成本昂贵，尤其是对于可编程芯片。因此，本发明的技术和机理允许子电路再用于两种类型的指令。

即使许多指令都是多字节的大小并且需要被对准在多字节地址上，本发明的技术和机理还是允许转移指令的单位是字节。

图 4 是允许字节对准的地址的转移指令的一个示意图。字段 401 和 403 保持为空。用于转移指令的操作码被包括在字段 407 中。转移偏移被包括在立即字段 405 中。然而，不同于常规的转移指令，立即字段值被以字节为单位保持而不是以字为单元。例如，常规指令将指示偏移的大小是 32 字。32 的值将被保持在常规转移指令的立即字段中。乘以 4 的乘法运算将指示偏移大小是 128 字节。根据各种不同的实施例，128 而不是值 32 被保持在立即字段 405 中。因为 128 的值被保持，所以不需要乘法运算或其它换算。通过将偏移符号扩充到指令地址中的比特数并将其加到转移指令地址或程序计数器，目标地址被计算出来。根据各种不同的实施例，符号扩充的偏移被加到程序计数器值的变型上，比如程序计数器加上 4。在一个实施例中，该计算如下所示：

转移目标=程序计数器+符号扩充(立即字段)

因为转移的单位是字节并且程序计数器值和转移目标的单位是字节，所以不需要乘以 4。

图 5 是典型的非转移指令的一个示意图。非转移指令包括源寄存器字段 501、目的寄存器字段 503、以及立即值字段 505、以及诸如整数加指令 507 之类的操作码。根据各种不同的实施例，整数加指令取源寄存器字段 501 所指定的寄存器值，将其加到立即字段 505 的符号扩充值上，并把计算结果写入目的寄存器字段 503 指定的寄存器。在一个实施例中，计算如下所示：

目的寄存器值=源寄存器值+符号扩充(立即字段)

因为转移和非转移操作都执行对立即字段值的符号扩充操作，所以相同的子电路可以被再用于两种类型的操作。使用相同的子电路允许有效利用硬件资

源并减少处理器核心的大小。

图6是用公共子电路来处理转移和非转移操作的一个技术流程图。在601，源寄存器值被获得。在603，用公共的子电路来执行对源寄存器值和立即字段值的操作。在一个例子中，所使用的公共子电路是用于执行符号扩充操作的电路。在605，计算结果被写入目的寄存器。在611，与程序计数器相关的值被获得。在613，用603中示出的相同子电路对程序计数器和立即字段值执行“与”操作。在615，在字节对准的地址上执行转移。应当指出，尽管图6被用来描述非转移和转移指令的含蓄的例子，但是许多其它的例子和配置也是可能的。此外，执行指令可能涉及诸如载入和存储之类的各种其它的操作，为清楚起见就不在此描述了。

尽管本发明的技术和机理可应用于各种不同处理器，包括通用处理器、视频加速器、密码学加速器、数字信号处理器、微控制器等等，然而本发明的技术和机理尤其可应用于可编程芯片和可编程芯片系统上的软核心处理器。可编程芯片上的逻辑往往是非常珍贵的，因此核心大小的减少往往是特别有益的。

图7是可用于实现本发明技术的可编程芯片上的一个系统实例的示意图。该系统包括处理器核心、硬件加速器、外围设备和外围接口。处理器核心大小可以通过再用子电路来执行转移和非转移操作而被优化。外围设备和外围接口在此称之为元件。可编程芯片上的系统包括处理器核心 701 和以太网元件 703 以及外部元件UART 711、PIO 713、定时器715和数据存储器717。在某些例子中，以太网元件103用数字信号处理(DSP)核心、密码学加速器或视频处理器来替代。应当指出，系统可以包括芯片内存储器 717 和芯片外存储器。在一个例子中，数据存储器 717 可以支持可变延时或固定延时访问。元件用互连架构707互连。用于连接系统中的元件的任何机理或逻辑在此称之为互连架构。在一个例子中，互连架构是总线。在另一个例子中，互连架构是从端仲裁架构。

可编程芯片使用不同类型的资源，其可以在不同分配中被互换地使用以便实现可编程芯片上的系统。在一个例子中，可编程芯片使用逻辑元件来实现可编程芯片上的每个不同的元件。

逻辑元件一般可以用诸如反熔丝、静态RAM和EPROM之类的元件来实现。可编程芯片上的任何机理在此称之为逻辑元件，可编程芯片基于编程信息对已知数量的输入线执行操作来提供一个或多个输出。一些逻辑元件被实现为

查阅表和用于执行输入线上的布尔操作的开关的组合。在一个例子中，逻辑元件包括能够实现任意 4 输入逻辑功能的 16 位 SRAM 查阅表(LUT)，形成快速进位链和快速级联链的电路、寄存器和用于寄存器的预置/重置逻辑。

本发明的技术和机理允许系统在来自高级语言程序的可编程芯片上的实现。在一个例子中，可变延时和固定延时可以在使用常规总线结构的系统上被支持。

系统总线一般具有设置宽度(例如 64 比特、128 比特)并且在任一时刻只允许一个主元件主动地使用总线。在常规系统中，在任何给定时间只有一个主元件可以访问给定从元件中的任何一个。以如下方式访问从元件的多个主元件在此称之为同时访问从元件，即，如果在相同的数据线上执行会导致数据比特冲突的方式。

在一个例子中，以太网元件正在访问从 PIO。当以太网元件正在访问从 PIO 时，即使主流输出设备和外围接口都是可用的，处理器也不能经由外围接口访问 SDRAM。

根据本发明的各种不同实施例，应当认识到在诸如芯片上的系统、可编程芯片上的系统，及其它计算机系统实现之类的某些应用中不再需要总线。使用硬件描述符语言(HDL)的诸如可编程逻辑器件(PLD)或现场可编程门阵列(FPGA)之类的器件在此称之为可编程芯片或可编程器件。替代实现复杂的总线共享方案而使用诸如拆分(splitting)之类的机理，总线本身可以被除去以提高系统性能。

根据特定实施例，应当认识到主元件和从元件不需要经由诸如总线之类的构造来布线。通过不经由总线来路由信号，流输出设备可以用更加有效的方式来实现。构成总线的常规数据和地址线不再是被争用的资源。因为在系统中连接到每个从元件的物理线数量可以被固定，所以代之以从元件成为争用的资源。此外，不使用总线增强了互连的灵活性。例如，以太网元件可以被分配各种端口来直接访问存储器，而存储器将成为唯一被争用的资源。

因此不再需要与计算机系统中所有的从元件相关联的系统总线仲裁器。作为替代，可以被多于一个主元件访问的从元件本身被分配个别的从端仲裁器。对应于可由多于一个主元件访问的特定从元件的仲裁器在此称之为从端仲裁器。在一个实施例中，在计算机系统中对于每个从元件都有从端仲裁器。在其它实施例中，有一个从端仲裁器用于系统中被选择的从元件。本发明的技术认

识到，对高级语言程序的以太网元件支持可以通过使用从端仲裁而更有效和实际地提供在系统中。

图 8 是使用从端仲裁的一个系统例子的示意图，有时被称为从属侧仲裁，同时使用多个主元件或同时使用多个主方(master)。使用对应于可由多于一个主元件访问的个别从元件的个别仲裁器的系统在此称之为从端仲裁系统。从端仲裁系统不再需要总线或系统总线仲裁器，当第一主元件正在访问第一从元件时，它们阻止第二主元件访问第二从元件。根据各种不同的实施例，诸如外围接口 825 之类的从元件与从端仲裁器 851 相关联。然而，从元件 UART 821 和 PIO 823 不与任何仲裁器相关联。在一个例子中，从元件 UART 821 和从 PIO 823 只能由主 CPU 813 访问而不能由主以太网设备 815 访问。然而，从存储器元件 825 可以被主 CPU 813 和主以太网设备 815 两者访问。

根据各种不同的实施例，从端仲裁器 851 允许第一主元件在第二主元件访问系统中的第二从元件的同时访问系统中的第一从元件。例如，主以太网 815 可以经由从端仲裁器 851 访问外围接口 825，同时主 CPU 813 访问从 UART 821。

通过允许 CPU 在诸如流输出设备或以太网元件之类的另一个主元件访问存储器的同时访问从元件，总线瓶颈可以被减少。通过使用同时的多个主元件结构还可以支持元件之间更直接的连接。

图 9 是可编程芯片上的系统实现的一个示意图。输入级 901 一般从用户接收选择信息以用于诸如处理器核心之类的逻辑以及诸如将在电子器件上实现的流输出设备之类的其它元件。在一个例子中，被接收的输入是高级语言程序的形式。发生器程序 905 创建逻辑描述并把该逻辑描述以及其它定制的逻辑提供给多种任何综合工具、布局和布线(place and route)程序、以及允许逻辑描述实现在电子器件上的逻辑配置工具。

在一个例子中，输入级 901 往往允许将被用于电子器件的元件选择和参数化。输入级 901 还允许可变或固定延时支持的配置。在某些例子中，提供给输入级的元件包括知识产权函数、强函数(megafunction)、和知识产权核心。输入级 901 可以是使用向导的图形用户界面以允许高效或方便地输入信息。输入级还可能是文本接口或读取诸如电子表格、数据库表或获取选择信息的示意图之类的数据文件的程序。输入级 901 产生一个输出，其包含被选择的各种不同模块的相关信息。

在典型的实现中，发生器程序 905 可以识别该选择并产生具有用于实现各种不同模块的信息的逻辑描述。发生器程序 905 可以是从用户输入的模块信息中创建诸如 Verilog、Abel、VHDL 和 AHDL 文件之类的 HDL 文件的 Perl 脚本。在一个例子中，发生器程序识别要加速的高级语言程序的一部分。其它代码被留下用于在处理器核心上执行。根据各种不同的实施例，发生器程序 905 识别指针并提供用于每个指针的端口。具有发生器程序能力的一个工具是可编程芯片内系统(SOPC)编制器，它可以从加州圣何塞的 Altera Corporation 获得。发生器程序 905 还向综合工具 907 提供信息以允许自动综合 HDL 文件。在某些例子中，逻辑描述直接由设计者提供。由用户选择的各种不同元件之间的挂钩(hookup)也由发生器程序互连。一些可得到的综合工具是可从俄勒冈州 Wilsonville 的 Mentor Graphics Corporation 获得的 Leonardo Spectrum 以及可从加州 Sunnyvale 的 Synplicity Corporation 获得的 Synplify。HDL 文件可以包括仅仅由综合工具可读的技术特定的代码。HDL 文件在此也可以被传送到仿真工具 909。

所属领域技术人员应当了解，输入级 901、发生器程序 905 和综合工具 907 可以是单独的程序。单独的程序之间的接口可以是数据库文件、日志、或仅仅是程序之间传送的消息。例如，替代于向存储器写入文件，输入级 901 可以直接向发生器程序 905 发送消息从而允许发生器程序创建逻辑描述。同样地，发生器程序可以直接向综合工具提供信息而不是写入 HDL 文件。同样地，输入级 901、发生器程序 905 以及综合工具 907 可以被集成到单个程序中。

用户可以选择各种不同的模块，集成的程序可以取得用户选择并以没有中间文件的综合后网表的形式来输出逻辑描述。用于描述将被实现在电子器件上的逻辑的任何机理在此称之为逻辑描述。根据各种不同的实施例，逻辑描述是诸如 VHDL、Abel、AHDL 或 Verilog 文件之类的 HDL 文件。逻辑描述可以在元件的用户选择和器件最终配置参数之间的处理的不同阶段中。根据其它实施例，逻辑描述是诸如电子设计交换格式输入文件(EDF 文件)之类的综合后网表。EDF 文件是综合后网表文件的一个例子，其可以由综合工具 907 输出。

综合工具 907 可以取得 HDL 文件并输出 EDF 文件。综合工具允许在电子器件上实现逻辑设计。一些可以获得的综合工具是可从俄勒冈州 Wilsonville 的 Mentor Graphics Corporation 获得的 Leonardo Spectrum 以及可从加州 Sunnyvale 的 Synplicity Corporation 获得的 Synplify。各种不同的综合后网表格式将被所属

领域技术人员所了解。

验证级 913 一般在综合级 907 之后。验证级检查设计的准确性以确保中间或最终设计实现预期的要求。验证级一般包括仿真工具和定时分析工具。仿真工具可以不必实现物理器件而允许输入应用和输出检查。对于设计的功能和定时验证，仿真工具向设计者提供成本效率高和有效的有效机理。功能验证涉及与定时考虑无关的电路逻辑操作。诸如门延迟之类的参数被忽略不计。

定时验证涉及分析具有定时延迟的设计操作。设置、保持、及其它用于诸如触发器之类的时序设备的定时要求被确认。一些可以获得的仿真工具包括 Synopsys VCS、VSS 和 Scirocco，它们可以从加州 Sunnyvale 的 Synopsys Corporation 获得，而 Cadence NC-Verilog 和 NC-VHDL 可以从加州圣何塞的 Cadence Design System 获得。在验证级 913 之后，被综合的网表文件可以被提供给包括布局和布线以及配置工具的物理设计工具 919。布局和布线工具一般在目标硬件器件的特定逻辑元件上定位逻辑单元，并且根据实现电子设计所需的逻辑连接各种逻辑元件的输入和输出之间的线路。该器件也可以在 923 被物理上地测试。

对于可编程逻辑器件，可编程序逻辑配置级可以使布局和布线工具的输出凭借用户选择和参数化的模块来编程逻辑器件。根据各种不同的实施例，布局和布线工具以及逻辑配置级在 Quartus 开发工具中被提供，它可以从加州圣何塞的 Altera Corporation 获得。所属领域技术人员应当了解，多种综合、布局和布线、以及可编程逻辑配置工具可以使用本发明的各种不同技术来测试。

上面提到，不同的级和程序可以被集成到各种方式中。根据一个实施例，输入级 901、发生器程序 905、综合工具 907、验证工具 913 以及物理设计工具 919 被集成到单个程序中。不同级自动运行并且对用户透明。程序可以接收用户选择的模块，产生描述用于实现各种不同选择模块的逻辑的逻辑描述，并实现电子器件。所属领域技术人员应当了解，HDL 文件和 EDF 文件仅仅是逻辑描述的例子。其他的文件格式以及内部程序表示是逻辑描述的其它例子。

图 10 说明了可被用于实现具有共享 I/O 线路的可编程芯片的典型计算机系统。计算机系统 1000 包括任意数量的处理器 1002(也称之为中央处理单元或 CPU)，它们被连接到包括存储器 1006(一般是随机存取存储器或“RAM”)、存储器 1004(一般是只读存储器或“ROM”)的设备。处理器 1002 可以被配置来产

生用于任何指定处理器的测试序列。在本技术领域中所熟知的是，存储器 1004 向 CPU 单向传送数据和指令，并且存储器 1006 一般被用来以双向方式传送数据和指令。

这两个存储器设备都可以包括上述的任何适当类型的计算机可读介质。此外，大容量存储设备 1008 被双向连接到 CPU 1002 并提供附加的数据存储容量，而且可以包括任何上述的计算机可读介质。大容量存储设备 1008 可以被用来存储程序、数据等等，并且一般是诸如慢于存储器的硬盘之类的从存储介质。大容量存储设备 1008 可以被用来保持预封装的逻辑或知识产权功能的程序库(library)或数据库，以及关于产生特定配置的信息。应当理解，保持在大容量存储设备 1008 内的信息在适当的情况中可以以标准方式作为虚拟存储器被结合为存储器 1006 的一部分。诸如 CD-ROM 1014 之类的特定的大容量存储设备还可以向 CPU 单向传递数据。

CPU 1002 还被连接到接口 1010，接口 1010 包括一个或多个输入输出设备，比如视频监视器、跟踪球、鼠标、键盘、麦克风、触敏显示器、传感器读卡机、磁带或纸带读带机、书写板、触针、语音或手写识别器、或其它熟知的输入设备，当然包括其它计算机。最后，CPU 1002 可以选择性地连接到使用通常如在 1012 所示的网络连接的计算机网或电信网。有了这类网络连接，在上述方法步骤的执行的过程中，期待 CPU 可以从网络接收信息或可以向网络输出信息。应当指出，系统 1000 还可以与把完成的设计传送到可编程芯片上的设备相关联。上述的设备和材料对于计算机硬件和软件领域的技术人员来说是很熟悉的。

上述的硬件元件可以被配置(通常临时地)来充当执行本发明操作的多个软件模块。例如，运行发生器程序的指令、输入级(例如向导)，和/或编译器可以被存储在大容量存储设备 1008 或 1014 上并结合主存储器 1006 运行在 CPU 1008 上。

尽管上述的许多元件和处理是为方便起见的单一形式，然而所属领域技术人员应当了解，多个元件和重复处理也可以被用来实践本发明的技术。

虽然本发明已经被示出并参考其特定实施例被说明，但是所属领域技术人员应当了解，公开实施例的形式和细节可以在没有背离本发明的精神或范围的前提下被改变。例如，本发明的实施例可以用多种的主次元件来工作并不应该受限于上述实施例。因此，本发明意在被解释为包括所有属于本发明真实精神和范围的变型和等效物。

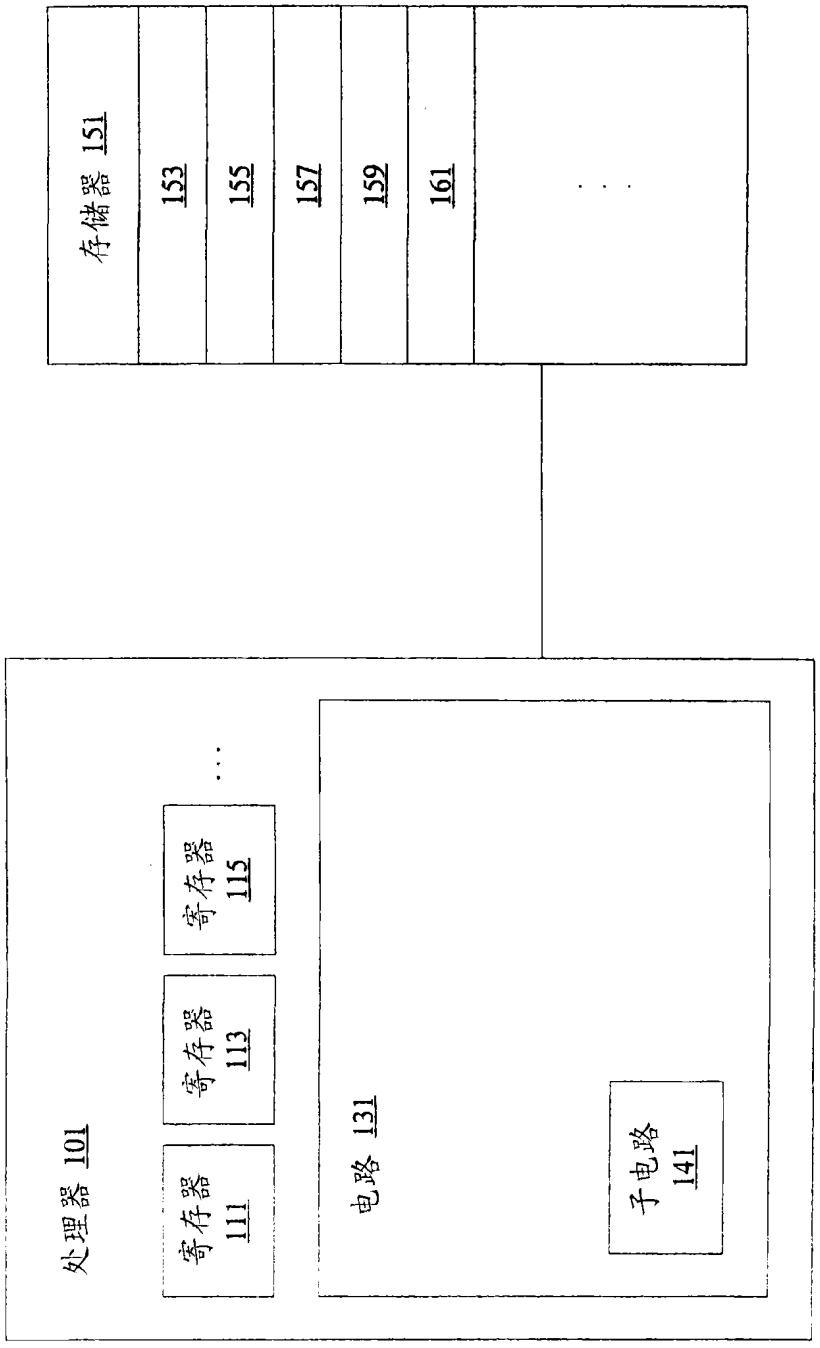

存储器 151
153
155
157
159
161
. . .
处理器 101
寄存器 111
寄存器 113
寄存器 115
. . .
电路 131
子电路 141

图 1

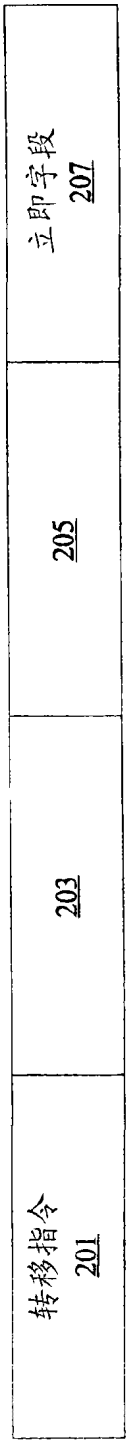



图 2

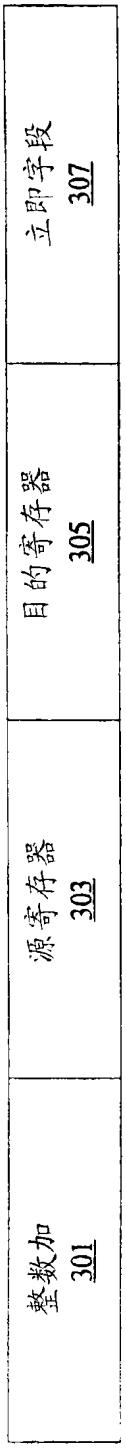



图 3

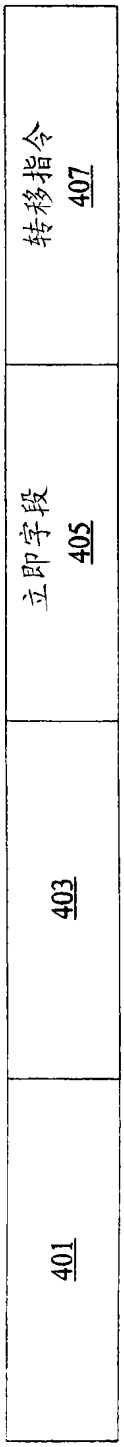



图 4

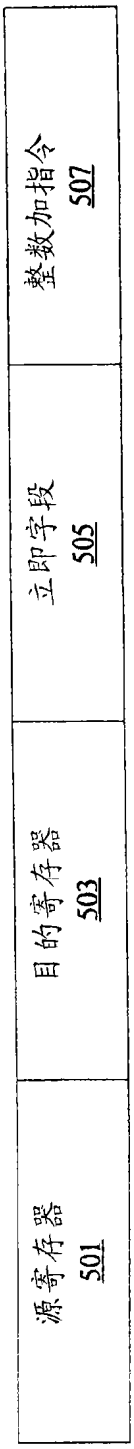



图 5

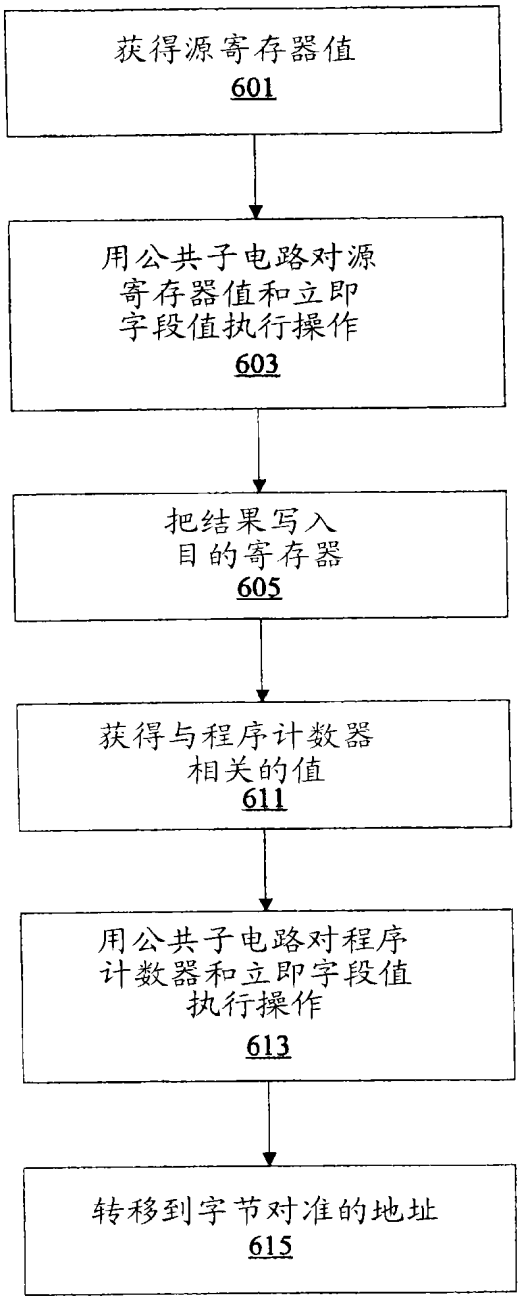

获得源寄存器值
601
用公共子电路对源寄存器值和立即字段值执行操作
603
把结果写入目的寄存器
605
获得与程序计数器相关的值
611
用公共子电路对程序计数器和立即字段值执行操作
613
转移到字节对准的地址
615

图 6

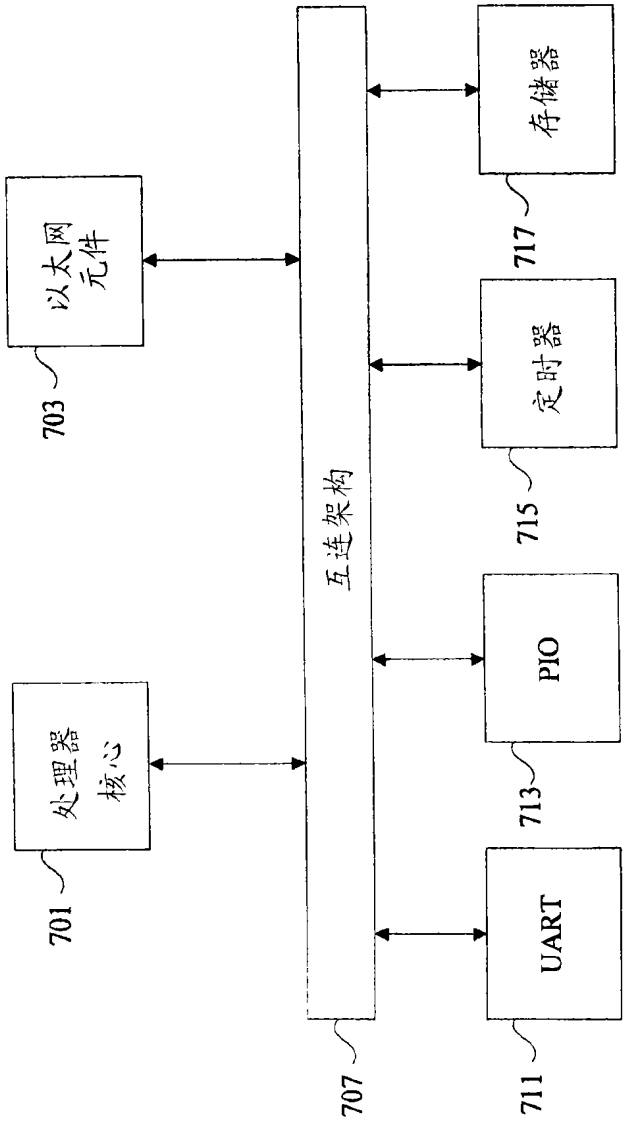



图 7

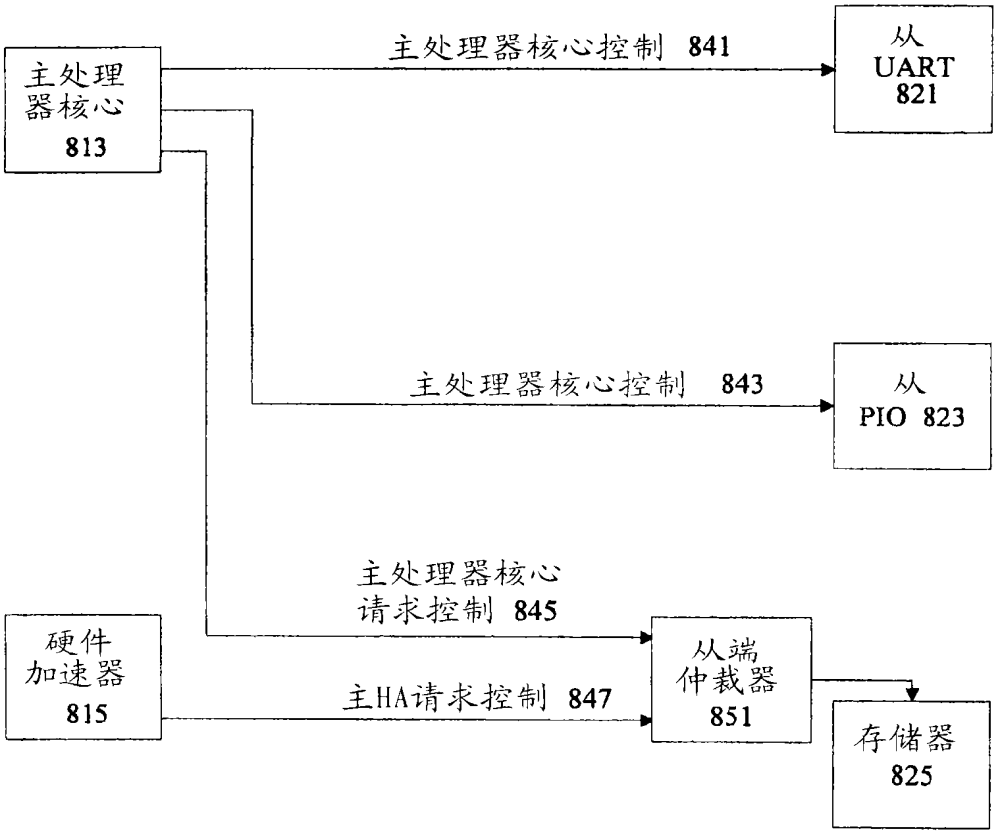

主处理器核心 813
主处理器核心控制 841
从 UART 821
主处理器核心控制 843
从 PIO 823
主处理器核心请求控制 845
硬件加速器 815
主HA请求控制 847
从端仲裁器 851
存储器 825

图 8

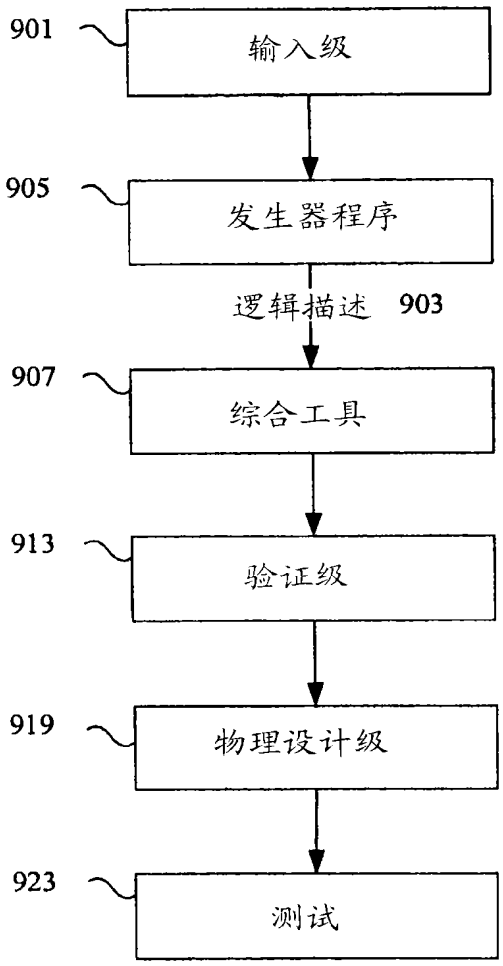



图 9

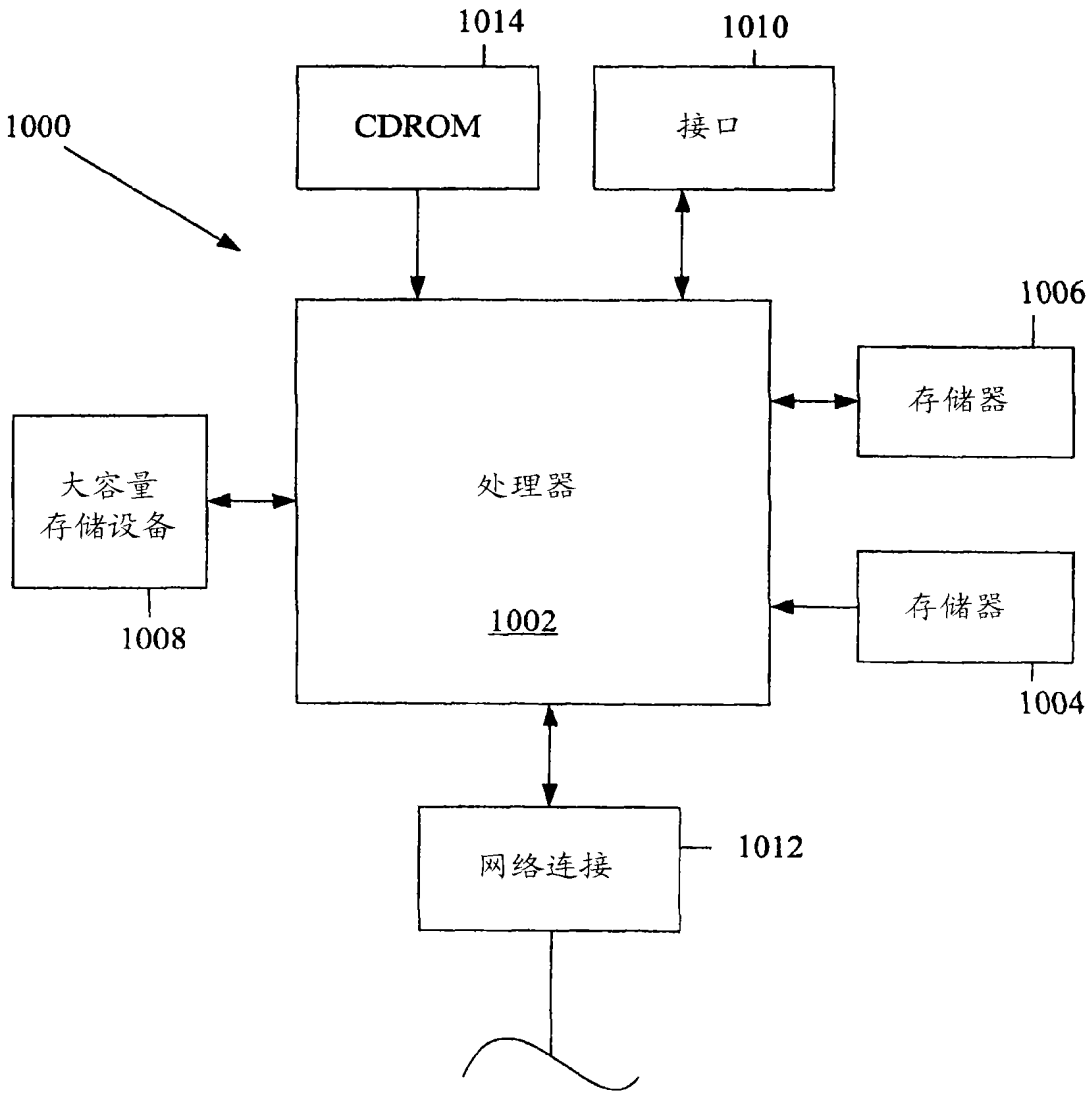

1014
1010
1000
CDROM
接口
1006
存储器
大容量
存储设备
处理器
1002
存储器
1008
1004
网络连接
1012

图 10